***Listing of the Claims*:**

1. (Currently Amended) A system for non-intrusively monitoring an application, comprising:

a first module ~~operable to~~ stored on a computer-readable medium that attaches to a memory area that is used by an application [[in]]during real-time operation, the first module ~~further operable to~~ reads application values from the memory area that have been stored in the memory area by the application [[in]]during real-time operation;

a second module stored on a computer-readable medium in communication with the first module ~~and operable to~~ that requests the first module to read the application values, the second module ~~further operable to~~ receives the application values from the first module; and

a third module stored on a computer-readable medium in communication with the second module~~, the third module operable to~~ that displays the application values.

2. (Original) The system of Claim 1, wherein the memory area is further defined as a shared memory of the application.

3. (Original) The system of Claim 1, wherein the first module is further operable to attach to the memory area used by the application to read the application values.

4. (Original) The system of Claim 1, wherein the application values are further defined as at least one application variable and a value for the application variable.

5. (Original) The system of Claim 1, wherein the first module is further operable to communicate the application values to the second module in hypertext markup language format.

6. (Original) The system of Claim 1, wherein the third module is further defined as a graphical user interface.

7. (Original) The system of Claim 6, wherein the graphical user interface is further operable to receive an input identifying the application values to be read and operable to request the application values identified to the first module, via the second module, and wherein the first module is operable to read the requested application values data from the memory area and return the application variables to the graphical user interface, via the second module.

8. (Original) The system of Claim 6, wherein the graphical user interface is further operable to receive an input identifying requested application values to be displayed.

9. (Original) The system of Claim 1, wherein the first module is further operable as a socket server and wherein the second module is further operable as a socket client such that the first and second modules communicate via a socket connection.

10. (Original) The system of Claim 1, wherein the first module operable to read application values stored in the memory area by the application while the application is running.

11. (Original) The system of Claim 10, wherein first module operable to read application values stored in the memory area by the application without interfering with the operation of the application.

12. (Currently Amended) A method of non-intrusively monitoring operation of an application, comprising:

running an application in a real-time manner;

generating, by the application, application values ~~stored in a memory area~~ during operation of the application;

storing, by the application, the application values in a memory area during the operation of the application;

reading, by a monitor, the memory area used by the application to obtain the application values, wherein at least one of the application values is not output by the application;

and

displaying the application values read from the memory area.

13. (Currently Amended) The method of Claim 12, further comprising:

requesting, by a client, application values from the[[a]] monitor ~~and wherein the monitor reads the memory area to obtain the application values~~; and

communicating the application variables from the monitor to the client.

14. (Currently Amended) The method of Claim 13, further comprising:

requesting application values;

running a plurality of applications in a real-time manner;

generating application values stored in one or more memory areas during operation of the plurality of applications;

reading the one or more memory areas used by the plurality of applications to obtain the application values; and

displaying the requested application values.

15. (Original) The method of Claim 14, wherein the memory area is further defined as a block of shared memory and wherein the monitor reads the at least some of the application variables stored in the block of shared memory

16. (Original) The method of Claim 13, further comprising providing memory manager and wherein the monitor registers with the memory manager to obtain a location of the memory area used by the application to store the application values.

17. (Original) The method of Claim 13, further comprising:

generating new application values by the application stored in the memory area, at least one of the new application values defined as a new value for a variable of the application;

requesting, by the client, that the monitor re-read the application values stored in the memory area;

re-reading, by the monitor, the memory area to obtain the new application values.

18. (Original) The method of Claim 17, wherein the monitor reads the application values while the application is running

19. (Original) The method of Claim 13, wherein the monitor is operable as a socket server and wherein the client is operable as a socket client such that the communication between the monitor and client is via a socket connection.

20. (Original) The method of Claim 12, wherein the application values are further defined as a variable of the application and a value of the variable.

21. (Currently Amended) A system for non-intrusively monitoring variables during operation of an application, comprising:

a compile listing stored on a computer-readable medium having an address map with an offset for at least one variable of an application; and

a module stored on a computer-readable medium ~~operable to~~ that reads the compile listing and obtains the offset of the at least one variable of the application, the module ~~further operable to~~ attaches to an address space ~~where~~ used by the application ~~is operating~~ during real-time operation to obtain a value for the variable during the real-time operation of the application using the offset.

22. (Original) The system of Claim 21, wherein the module is further operable to read the compile listing and convert the variable to the offset.

23. (Original) The system of Claim 21, wherein the module is further operable to search the compile listing and display a plurality of variables of the application for selection by a user.

24. (Original) The system of Claim 23, wherein the module is responsive to selection by the user of one of the plurality of variables to obtain the value for the selected one of the plurality of variables using the offset to locate the value of the variable in the address space.

25. (Original) The system of Claim 24, wherein the module is further operable to display the selected one of the plurality of variables.

26. (Original) The system of Claim 21, wherein the address space is further defined as a memory space and wherein the module attaches, using a socket layer, to the memory space used by the application.

27. (Original) The system of Claim 26, wherein the module attaches, using the offset, to the memory space used by the application via an operating system service.

28. (Original) The system of Claim 21, wherein the monitor is further operable, using the compile listing, to query the address map for one or more of the variables of the application.

29. (Original) The system of Claim 21, wherein the module is further defined as a subtask of the operating system.

30. (Original) The system of Claim 21, wherein the module is further operable to attach to the memory space where the application is operating and overwrite the value for the variable using the offset.

31. (Original) The system of Claim 21, wherein the module comprises:

a reader component operable to read the compile listing and further operable to convert the at least one variable of the application to the offset; and

a search component receiving the offset of the at least one variable from the reader component, the search component operable to attach to the application and further operable to locate the value of the variable using the offset.

32. (Original) The system of Claim 21, further comprising display component operably coupled to the module to receive the value of the variable, the display component operable to display the value.

33. (Original) The system of Claim 32, wherein the display component is operable to employ the value to display a heartbeat.

34. (Original) The system of Claim 32, wherein the display component is operable to employ the value to display as a percentage complete.

35. (Currently Amended) A system for non-intrusively monitoring COBOL application values, the system comprising:

a memory area;

a COBOL program stored on a computer-readable medium ~~operable to~~ that generates program values and stores the program values in the memory area during real-time operation of the COBOL program; and

a COBOL monitor module stored on a computer-readable medium ~~operable to~~ that shares the memory area with the COBOL program [[to]]and reads the program values stored in the memory area by the COBOL program during real-time operation of the COBOL program.

36. (Original) The system of Claim 35, further comprising:

a second COBOL program operable to generate second program values and store the program values in the memory area during real-time operation of the second COBOL program, and wherein the COBOL monitor module is further operable to read the second program values stored in the memory area by the second COBOL program.

37. (Original) The system of Claim 35, further comprising:

a second memory area; and

a second COBOL program operable to generate second program values and store the program values in the second memory area during real-time operation of the second COBOL program, and wherein the COBOL monitor module is further operable to read the second program values stored in the second memory area by the second COBOL program.

38. (Original) The system of Claim 35, further comprising:

a user interface operable to monitor and display the application values; and

a client application in communication with the user interface and the COBOL monitor module, the client application operable to request the program variables of the COBOL program from the COBOL monitor module and provide the program variables to the user interface for display via the user interface responsive to a request from the user interface.